

<title><b>peer to peer</b></title>



using xml and xsl to generate web pages  
by narti kitiyakara > >



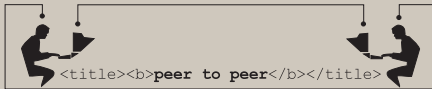
first programmer or fairy lady?  
by danielle deases > >

still writing the old fashioned way?  
10 tips for getting up to date.  
by suzan st. maur > >



  
**NOLA**  
COMPUTER  
SERVICES

ISO 9001 Registered



We publish **peer to peer** to recognize the accomplishments of our NOLA team members and to provide our customers with information about the field of IT.

We offer a broad range of IT services to the public and private sectors. Our services include information management consulting, Internet/intranet/extranet development, system architecture, and project management.

For inquiries, please call us toll free at 888.488.1101 or visit us at [www.nolacom.com](http://www.nolacom.com).

**peer to peer thanks**

Suzan St. Maur and MarketingProfs.com—a provider of strategic and tactical marketing know-how—for kind permission to reprint Ms. St. Maur's "10 Online Writing Concepts."

**managing editor**

Trish Thomas

**contributors**

Syed Baber  
Danielle DeAses  
Narti Kitiyakara  
Suzan St. Maur

**proofreading**

Danielle DeAses

**design**

Trish Thomas

We welcome your comments at [peertopeer@nolacom.com](mailto:peertopeer@nolacom.com).

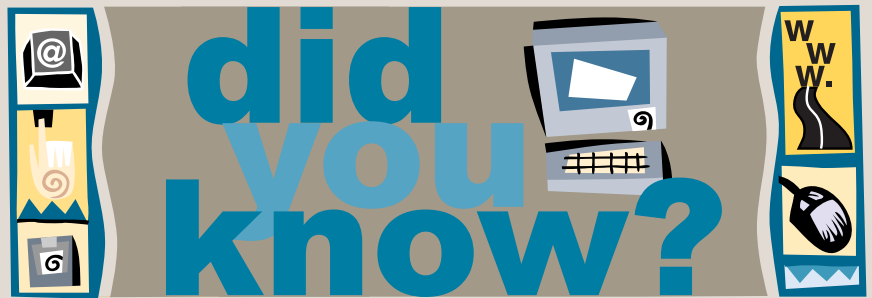
**Atlanta**  
**Houston**  
**New Orleans**  
**Washington D.C.**

**Corporate Headquarters**

3535 Canal Street  
New Orleans, Louisiana 70119  
Voice: 504.488.1111  
Facsimile: 504.488.9955  
[www.nolacom.com](http://www.nolacom.com)



iso 9001 registered



**Two different father- and son-owned companies, Sperry**

Gyroscope and Sperry Aircraft Company, merged in 1933 to become Sperry Corporation. Sperry specialized in such high-tech devices as computer controlled bombsights, airborne radar systems, and automated take off and landing systems, which proved to be quite profitable during World War II.

Another merger around the same time produced Remington Rand Corporation, combining Remington Typewriter and Rand Kardex. Remington Rand's specialty was typewriters, adding machines, and other office equipment.

Hook up a specialty in high-tech control systems with a specialty in business machines and a focus shift to computers is the logical next step. That's just what happened when Sperry and Remington Rand combined to form Sperry Rand Corporation in 1955. A new company, a new focus, and some important men in the upper echelon—General Leslie R. Groves, who had headed the Manhattan Project that developed the atomic bomb, and General Douglas MacArthur, famous for leadership in the Pacific during World War II and for being fired by President Harry Truman during the Korean Conflict.

Sperry Rand was one of the first companies to begin manufacturing computers—they developed the UNIVersal Automatic Computer series. The first UNIVAC was delivered to the census bureau in 1951, followed by UNIVAC II in 1955. Larger than refrigerators, they ran off of punched cards. Several updated versions followed, but eventually UNIVACs were cast off for more refined systems.

Although "Mac" was mostly a figurehead, he did create a few legends in the company. The senior officers had gotten into the habit of drifting in around 9 in the morning. When Gen. MacArthur discovered this, he did something about it. For several weeks, he would show up at 8 and stand in the doorway across the street, watching for the VPs to check in. They got the message.

General MacArthur also enjoyed having his picture taken, with everyone. One Sperry Rand exec remembers being photographed standing on the floor while a little platform was provided for the General. Although said to be about 6 feet tall, General MacArthur was not quite as tall as the exec, and apparently the General looked up to no one.

Sperry Rand was eventually absorbed into the Unisys Corporation of Pennsylvania.

Come see us at  
**XP Agile Universe.**

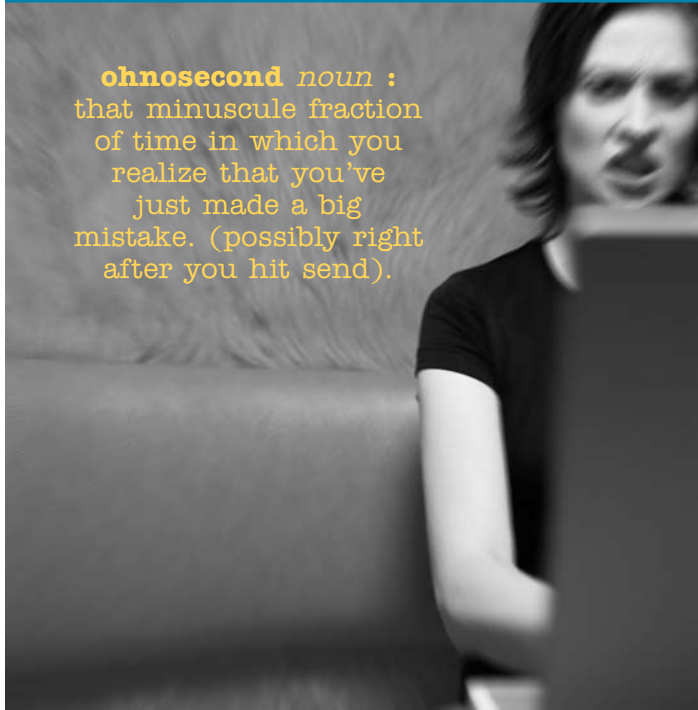
For complete information visit [www.agileuniverse.com](http://www.agileuniverse.com).

# signs of e-times

## address munging

*present participle* : altering your return e-mail address in an effort to thwart spammers. from *to munge*, as in to destroy or make useless.

**ohnosecond** *noun* : that minuscule fraction of time in which you realize that you've just made a big mistake. (possibly right after you hit send).



New times. New technologies. New words.

During the 1940s, Random House added spaceship, tape recorder, and A bomb to their dictionary (also cheeseburger and bikini). In the 50s they added aerospace, computerize, and data processing (along with hash browns, bermuda shorts, and weirdo). In the 60s we got cable television, jet lag, instant replays, megabytes, and pantsuits, and the 70s gave us personal computers and diskettes (also leg warmers and trail mix). Virtual reality, automated-teller machines, and compact discs came in the 80s and, in the 90s, we got Web sites, V-chips, scrunchies, and buffalo wings. What have we gotten recently? Read the words words words.

**whitelist** *verb* : to place a name, e-mail address, Web site address, or program on a list of items that are deemed spam- or virus-free.



## blamestorming

*present participle* : sitting around in a group discussing why a deadline was missed or project failed, and who was responsible.



**mouse potato** *noun* : e-times answer to the couch potato.

## Percussive maintenance

*noun* : the fine art of whacking the cr\*\* out of an electronic device to get it to work again.

## seen any signs?

we'd like to hear about them and include them in future issues of **peer to peer**. cite your sitings at [peertopeer@nolacom.com](mailto:peertopeer@nolacom.com).



# first programmer or fairy lady?

ddeases@nolacom.com

danielle deases recently received her b.a. in psychology from the university of dallas. when she's not hard at work, she's using her newfound free time to catch up on her reading.



## “She walks in beauty like the night...”

No, Lord Byron wasn't referring to his daughter here, but she has been called the most picturesque figure in the history of computers.

Born on December 10, 1815, Augusta Ada Byron was whisked away to London by her mother just five weeks later. Soon after, Lord Byron left England for Greece, never to return alive or see his daughter again.

Determined that Ada would not be a poet like her father, Lady Byron hired a series of tutors to train Ada in mathematics, science, and music—but she couldn't overcome heredity. Ada's understanding of mathematics seems, indeed, to have been poetic. Her writings, particularly her letters to Charles Babbage expressing her ideas for his Analytical Engine, include a great deal of fantasy and metaphor.

When she was 17, Ada was

introduced to Mary Somerville, a Scottish scientist whose theories had led to the discovery of the planet Neptune. Ms. Somerville was one of Ada's most influential tutors and taught her a great deal. It was at a dinner party at Ms. Somerville's home that Ada was introduced to Charles Babbage and the Analytical Engine. Although Mr. Babbage felt the Analytical Engine was a more sophisticated design than his earlier invention, the Difference Engine, his sponsors in Parliament were unwilling to fund a second project while the first was still incomplete, so he appealed to friends at home and abroad, eventually gaining private sponsorship.

Ada, according to biographer Betty Toole, was inspired by the “universality of his ideas,” though few others were. She began a long-term correspondence with Mr. Babbage that

greatly benefited both of them. Her passion for his ideas, especially his theory that one might be able to build a machine that could not only make predictions but also act on those predictions, motivated her to make great contributions to the field of computer science. Ada, according to a letter he sent to her, became his “Enchantress of Numbers.”

## In 1835, Ada married William King,

who soon inherited a noble title, making them the Earl and Countess of Lovelace. Lady Lovelace's rich social and academic life included such other notable figures as Charles Wheatstone, Charles Dickens, Michael Faraday, and David Brewster, the inventor of the kaleidoscope.

Charles Babbage continued to work on plans for his new Analytical Engine, and reported >> page 8

## In 1974, groups from each military branch independently proposed the development of a

Lt. Col. Bill Whitaker, with the Department of Defense for Research and Engineering, felt that a joint effort would be the best approach for the development of a standardized, high-order language to be used in all of their mission-critical projects. Until then, the DoD had been using hundreds of different specific languages, and often software projects could not be reused or were never even completed. A standard language would reduce the cost of development and maintenance of the software systems on which they were becoming increasingly reliant. So, in 1975 the DoD formed the High-Order Language Working Group (HOLWG) with Lt. Col. Whitaker as chair.

The purpose of the HOLWG was not to develop the language, but to determine requirements for a common language and to make recommendations for adoption and implementation. *Strawman* comprised the first set of technical requirements from the HOLWG. Strawman stated widely accepted and general goals for the language, such as reliability, efficiency, readability, and simplicity, which were subsequently refined in iterations called Woodenman, Tinman, and Ironman. Each iteration was periodically open to the public for comment. In 1977, twenty-three existing languages were evaluated against the Ironman >> page 9

common high-order language.

# 10 Online Writing Concepts

that work wonders for your offline marcomms, too.

by Suzan St. Maur

## Powerwriting:

The Hidden Skills  
You Need to Transform  
Your Business Writing

by Suzan St. Maur

Suzan St Maur's *Powerwriting* (Financial Times Management) shows that—when it comes to effective business writing—the actual craft of writing is only half the story.

The other half is knowing how to approach the exercise to begin

with—how to structure your message, how to understand your audience, and how to marry the two. To get that wrong is expensive, time-consuming, and professionally embarrassing. Yet, posits Ms. St

Maur, millions of dollars are wasted every year on business communications that don't work because the approach to the exercise—rather than the writing or design—is wrong.

*Powerwriting* shows how to think before you write, and how to get your message to work for the audience you need to address. It can also show you how to write that message so it gets the results you want, every time.

Canadian-born Suzan St Maur is a prolific business writer who has published such business communications books as *Writing Words That Sell* (Mercury Books) and *Writing Your Own Scripts and Speeches* (McGraw Hill).

After all the agonies we suffered some years ago when some tried to make offline text work online, we've finally turned the tables. Now we can borrow back a number of online writing concepts and use them to sharpen up our paper-based marketing communications.

Remember how early Web site text could make you cringe? Squinting at all 2000 solidly crammed words so obviously lifted straight from an equally cringe-making corporate brochure? Peering at that fat, uniformly gray column of garbage scrolling hypnotically up through the browser window?

Well, nearly all of that has gone to the Great Delete Tab in the sky, thanks to people like Jakob Nielsen (and many others) who showed us how to get real and write for the Web as it should be done.

## What Goes Around, Comes Around

Now, though, there's evidence that Dr. Nielsen's chickens are coming home to roost back in the old offline barn.

First of all, there's a cosmetic trend for online notions to powder the nose of paper-based communication... Web and e-mail jargon, smiley faces, text abbreviations (U h8 txt 2?), and more are turning up in printed material every day.

More usefully, many of us who write for a living are applying some online writing techniques and approaches to our offline work, too. In fact, the very "fashionableness" of all things online has given us the excuse we needed to clear out a lot of the

... printed documents have gotten away with being chinless and indecisive in the past, but no more ...

awful old junk that's been cluttering up some clients' offline text for years.

## Online to Offline: Key Connections

### 1. It's essential to have clear objectives.

Any piece of online communication that doesn't have clear-cut objectives comes over as chinless and indecisive. Many printed documents have gotten away with being chinless and indecisive in the past, but no more—possibly due, in part, to online influences. If they're going to be taken seriously today, printed comms need clear objectives too—driven by what you want to achieve, not just what you want to say.

### 2. People often prefer to scan and go back to get detail later.

Although online text has championed scanning, people have been scanning offline text like brochure copy since long before the WWW came to be. Online, to facilitate scanning we break up text with highlighting, bold type, and crossheads which enable readers to get the gist of our message in a

>> page 10

# DECOUPLING WEB PAGES USING XML & XSL TO GENERATE PAGES

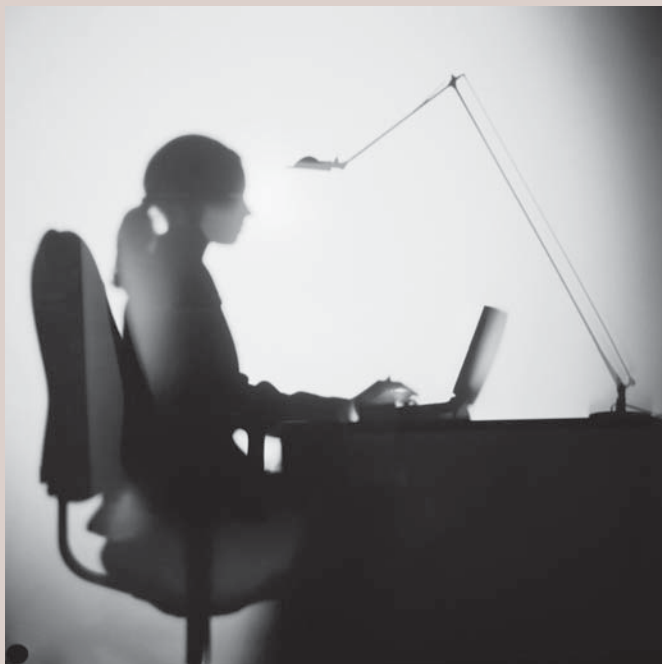
## WE HAVE LOOKED AT THE

problems associated with decoupling presentation from logic in the context of HTML pages (*peer to peer*, Summer 2001 and Winter 2003). In our first article we examined the problems of generating HTML pages from such scripting languages as ASP and JSP and concluded that neither of these technologies really lend themselves to decoupling presentation from business logic. In our second article we focused on such embedded tag technologies as ISAPI applications and custom tags on JSPs; these provide a clear advantage over the scripting languages in terms of decoupling and understandability.

Now we'll look at two more technologies that allow us to dramatically separate the logic of preparing our data from the presentation of that data: XML and XSL.

## THE HYPE SURROUNDING XML—EXTENSIBLE MARKUP

Language—always used to puzzle me. Sure, I saw the potential for a system-independent data representation, but—since I generally dealt with applications in a homogenous environment—there wasn't a lot of need for something that seemed to be



designed to deal with the problems of data distribution in a heterogeneous environment. Then I discovered XSL—eXtensible Stylesheet Language—and I got really excited. XSL, or, more properly, XSLT (the T is for Transformations) allows you to convert XML into HTML, XML in a different schema, or even PDFs. The obvious implication of this is that you can achieve a complete decoupling of HTML presentation from the logic that generates the data being presented. It was exactly what I was looking for.

Let's start out with a brief discussion about using XSLTs to create HTML documents. (For a more complete treatment, I recommend Neil Bradley's *The XSL Companion* published by Addison-Wesley.)

The basic building block of an XSL stylesheet is a template:

```
<xsl:template match="EducationItem">
  <tr>
    <td>
      <xsl:value-of
select="Institution"/>
    </td>
    <td>
      <xsl:value-of select="Degree"/>
    </td>
    <td>
      <xsl:value-of select="Year"/>
    </td>
  </tr>
</xsl:template>
```

In this example, whenever the XSL processor encounters an `EducationItem` element, it will output the HTML specified within the template. The `xsl:value-of` elements get the text of the elements specified by the "select" attribute. The "select" attribute can be any valid XPath expression, so you can refer to anything in your XML document. For example, if we were to transform the data file:

```
<EducationItem>
  <Institution>University of New Orleans</
Institution>
  <Degree>Master of Computer Science</
Degree>
  <Year>1984</Year>
</EducationItem>
```

by a stylesheet containing the above template, you'd end up with the HTML shown below.

```
<tr><td>University of New Orleans</
td><td>Master of Computer Science</
td><td>1984</td>
```

You can create as many templates as you need in a given stylesheet. For example, if you wanted to make the contents of the

Visit Narti  
and other NOLA  
XPers at  
XP Agile Universe  
August 10–13 in  
New Orleans.

after more than fifteen years of programming in many different languages, narti kitiyakara began coaching our XP development team. in this capacity he has become more interested in software process improvement in general and acceptance testing in particular. he introduced his acceptance testing system, avignon, at XP agile universe 2002.



[nkitiyakara@nolacom.com](mailto:nkitiyakara@nolacom.com)

Degree element bold, you could use two templates:

```
<xsl:template match="EducationItem">
  <tr>
    <td>
      <xsl:value-of
select="Institution"/>
    </td>
    <td>
      <xsl:apply-templates select="./
Degree"/>
    </td>
    <td>
      <xsl:value-of select="Year"/>
    </td>
  </tr>
</xsl:template>
<xsl:template select="Degree">
  <b><xsl:value-of select="."/></b>
</xsl:template>
```

Of course, you could also have done this by putting the `<b>` tags directly into the original template, but using this way lets you reuse the formatting of the Degree element in other places.

**XSL IS TOO BROAD A SUBJECT TO COVER HERE, BUT SOME OF THE MORE** interesting features of stylesheets are:

**CHOICES.** XSL allows you to vary your output with conditional statements. For example, we could automatically put “unknown” in the “year” column above by putting this fragment in the `<td></td>`:

```
<xsl:choose>
  <xsl:when test="count(Year)=0">unknown</
xsl:when>
  <xsl:otherwise><xsl:value-of
select="Year"/></xsl:otherwise>
</xsl:choose>
```

**LOOPS.** Actually loops don’t give you anything that templates can’t, but loops *can* make your stylesheets more concise. In the example above, you could loop over the EducationItems rather than making a template for them:

```
<xsl:for-each select="EducationItem">
  <!-- Same sort of stuff you'd have in
EducationItem template -->
</xsl:for-each>
```

**NUMBERING.** The `<xsl:number>` element allows you to insert item numbers into your output. You could, for example, number each of the EducationItem elements from the above example with Roman numerals by putting `<td><xsl:number format="i"/></td>` into the template. The default format is Arabic numerals, although you can use Arabic numerals, and Roman numerals, and letters. You can also control which number gets put where, whether or not elements matching a given pattern are numbered.

**SORTING.** The `<xsl:sort>` element allows you to change the order of elements in your output. For example, if we get a list of EducationItems that’s sorted by institution and we want to output it by descending year, we could do something like:

```
<xsl:for-each
select="EducationItem">
  <xsl:sort order="descending"
select="Year" data-type="numeric"/>
  <!-- Same sort of stuff you'd
have in EducationItem template -->
</xsl:for-each>
```

**SO, HAVING SEEN SOME OF THE THINGS WE CAN DO WITH** XSLT, it’s time to look at how we can actually perform these transformations in our code.

In the Java world, XML and XSL functionality is accessed through JAXP (Java API for XML Processing). This comes with J2SE 1.4 or you can download it separately from <http://java.sun.com/xml/jaxp/index.html> for earlier versions of the J2SE. JAXP gives you an interface to multiple implementations of the XML processing functionality. You plug an actual implementation into your application by specifying a system property on your java command line or by putting a `jaxp.properties` file into your `$JAVA_HOME/jre/lib` directory or a couple of other ways. This allows you to, for example, choose a processor optimized for speed or an entirely different one optimized for memory usage without having to change your own code.

To do a transformation in Java, you’d start out by getting an instance of a TransformerFactory and then call its new Transformer method. (You can set a number of properties on both the TransformerFactory and the Transformer, but I’ve never had a need to do so.) Once you have an instance of a Transformer, you can transform an XML document by using its transform method. This method takes an instance of a `javax.xml.transform.Source` and an instance of a `javax.xml.transform.Result`. The use of the Source and Result classes allow you to use the same transform method for any type of input and output. For example, [>> back cover](#)

on his efforts at a seminar at the University of Turin, Italy, in 1842. An Italian mathematician named Menabrea wrote his own description of the machine and published it in French the same year. Lady Lovelace translated Menabrea's paper into English and annotated the translation with her suggestions for the Engine's use.

Reading her famous predictions now one can see just how much ahead of her time Lady Lovelace really was. One of her most famous ideas combines her love for science with her love for music:

"Supposing, for instance, that the fundamental relations of pitched sounds in the science of harmony and of musical composition were susceptible of such expression and adaptations, the [analytical] engine might compose elaborate and scientific pieces of music of any degree of complexity or extent."

Remember the article in our last issue about the music-searching software called OMRAS? Could Lady Lovelace have predicted the OMRAS project nearly two hundred years before its conception?

She also presented a plan for using punched cards to instruct Mr. Babbage's Difference Engine to perform operations. Frenchman Joseph-Marie Jacquard had originally designed punched cards to *program* his looms (c. 1801), which produced the embroidered fabric that became his namesake. According to his design, patterns of holes in thin metal cards indicated where particular stitches should be placed. The design was partly based on the idea of a binary system, with a hole

meaning "1", or a stitch, and no hole meaning "0", or no stitch, making it easily applicable to computer programming.

Lady Lovelace maintained that one could use punched cards to input specific data and have Mr. Babbage's machine calculate Bernoulli numbers. The idea of using Jacquard's punched cards to store information for a calculating machine seems to be hers alone, and in this she anticipated by half a century the first application of this idea by Herman Hollerith for the U.S. census in 1890. Hollerith, founder of the company that would eventually become IBM, patented his own design of the cards, which was to become an amazingly successful way to program early computers.

Charles Babbage wrote about Lady Lovelace's ideas in his 1864 memoir, *Passages from the Life of a Philosopher*:

"Some time after the appearance of [Menabrea's] memoir on the subject in the 'Bibliothèque Universelle de Genève,' the...Countess of Lovelace informed me that she had translated the memoir of Menabrea. I asked why she had not herself written an original paper on a subject with which she was so intimately acquainted? To this Lady Lovelace replied that the thought had not occurred to her. I then suggested that she should add some notes to Menabrea's memoir...The selection [of illustrations] was entirely her own. So also was the algebraic working out of the different problems, except, indeed, that relating to the numbers of Bernoulli, which I had offered to do to save Lady

Lovelace the trouble. This she sent back to me for an amendment, having detected a grave mistake which I had made in the process...The Countess of Lovelace...has entered fully into almost all the very difficult and abstract questions connected with the subject."

Many now regard Lady Lovelace's plan as the first computer program. Speaking of her punched-card plan, the engineers at Penn State give her credit on their Web site: "From this idea of a program, written to run a computer, many programming languages were born."

On the other hand, the University of Exeter's "Babbage Pages" offer quite a different opinion:

"It is often suggested that Ada was the world's first programmer. This is nonsense: Babbage was, if programmer is the right term. After Babbage came a mathematical assistant of his, Babbage's eldest son, Herschel, and possibly Babbage's two younger sons. Ada was probably the fourth, fifth, or six [*sic*] person to write the programmes. Moreover all she did was rework some calculations Babbage had carried out years earlier. Ada's calculations were student exercises. Ada Lovelace figures in the history of the Calculating Engines as Babbage's interpretress, his 'fairy lady'. As such her achievement was remarkable."

Whomever you side with, Lady Lovelace's achievements are quite remarkable. Her additions to Menabrea's paper were published in Richard Taylor's *Scientific Memoirs Volume 3* in 1843. Unfortunately, she was unable to do much >> page 11

requirements. None satisfied the requirements but a few could be modified sufficiently. HOLWG decided to hold a design competition, and companies around the world contributed their designs. Teams

Navy  
Commander  
John Cooper  
suggested  
the name  
Ada,  
after the  
world's  
first  
computer  
programmer.

were color-coded to maintain anonymity for the duration of the contest.

The four finalists were Green: Cii Honeywell-Bull led by Jean Ichbiah; Red: Intermetrics led by Benjamin Brosgol; Blue: SofTech by John Goodenough; and Yellow: SRI International led by Jay Spitz. All four final languages were Pascal-based. Evaluations were rigorous and included an extensive written exam for each of the team leaders. The requirements were further refined to Steelman, and the top two, Green and Red, were compared to the new requirements. Finally the design from the Green team was selected.

Over all the years of its development, the >> page 11

puzzle guy syed baber is the customer support product manager in the commercial applications group. when he is not dreaming up ideas for new puzzles he is perfecting his (anti)hacking skills.



sbaber@nolacom.com

## Congratulations to Tony Cruze . . .

He solved the last puzzle correctly and was lucky enough to have his name drawn from a total of 5 correct responses. A computer support specialist at DOE in Oak Ridge, TN, Mr. Cruze received a \$50 gift certificate for lunch at his chosen restaurant, Red Lobster. Mr. Cruze correctly translated the IM conversation between Chatten' Charlie and Emoticon Ed to standard English. As with all translations, the interpretations we received varied slightly; Mr. Cruze's version is below. Correct answers were also submitted by Bruce Woods, Jay Shannon, Jennifer Kidder, and Savita Pradeep.

Speaker	IM	English
CC:	sup	What's Up?
EE:	hi	Hello.
CC:	Do u have the code?	Do You have the code?
EE:	:-S	Are you incoherent?
CC:	AWGHTGTGTTA	Are We Going To Have To Go Through This Again?
EE:	:-&	Tongue Tied.
CC:	FOAF told me	Friend Of A Friend Told Me.
EE:	\$__\$	Greedy.
CC:	GAL	Get A Life.
EE:	>-)	Devilish wink.
CC:	IIIO	Intel Inside, Idiot Outside.
EE:	:-]	Blockhead.
CC:	IOH	I'm Outta Here.
EE:	>-)	Devilish wink.

## . . . and now for the new puzzler.

Often when you visit a Web site that's based outside the U.S. you'll see an additional level in the URL indicating the country. For instance, [www.bbc.co.uk](http://www.bbc.co.uk) for a Web site based in United Kingdom.

Can you match the identifiers on the left to the countries on the right?

- |         |                     |
|---------|---------------------|
| 1. .es  | A. Iceland          |
| 2. .cx  | B. Christmas Island |
| 3. .is  | C. Spain            |
| 4. .de  | D. Germany          |
| 5. .il  | F. Israel           |
| 6. .li  | G. Czech Republic   |
| 7. .ms  | H. Montserrat       |
| 8. .au  | I. Australia        |
| 9. .at  | J. Austria          |
| 10. .cz | K. Liechtenstein    |



All readers are eligible to win a free lunch worth \$50. You don't have to be a NOLA team member. Just send a fax or e-mail to Puzzle Guy Syed (504.488.9955 or [sbaber@nolacom.com](mailto:sbaber@nolacom.com)) with your correct answer by July 31 and we'll include your name in a drawing that will determine the winner.

few seconds. Paper-based messages can be improved dramatically when given the same treatment.

3. **People do not always read in a linear fashion.** We don't expect people to view our Web site pages in any particular sequence. This is not new. For years people have been leafing through brochures starting at the back, skipping to the front, dipping into the middle and back again. Longish offline content benefits greatly from being organized on a non-linear basis to cater equally to the linear readers and the grasshoppers.
4. **Not everyone needs or wants the technical stuff.** Even with high-tech business, we often put the techie details in their own little cubby-hole on a Web site, or in a downloadable PDF file. That way they're there for those who are interested but don't obscure the main marketing messages. Offline messages gain in the same way, when you box off technical data or append it to the back of a document.
5. **Visual clutter confuses readers.** In the same way that people loathe Web site home pages that bristle with shouting headlines and graphics and other grinning gargoyles, they hate cluttered print and press ads that shriek "busy, busy." If it's hard to find your message in amongst garish junk, online or offline, they'll just flip or click over to your competitors' information.
6. **BS is boring. Everyone sees through hype now.** The online environment makes it look even sillier than ever. Readers of any marketing communication, online or off, expect your writing to talk directly to them, as one human being speaks to another. If you wouldn't insult a customer by using boastful, pompous hype face-to-face and online, why do it offline?
7. **Complex thinking doesn't work.** Although the long copy often works online, the writing style itself needs to be very economical and uncomplicated. Every word has to earn its keep. Sentences and paragraphs should be short and free from convoluted notions. And that's an approach that also works wonders to clarify and enliven text for brochures, print newsletters, and other longer marcomms.



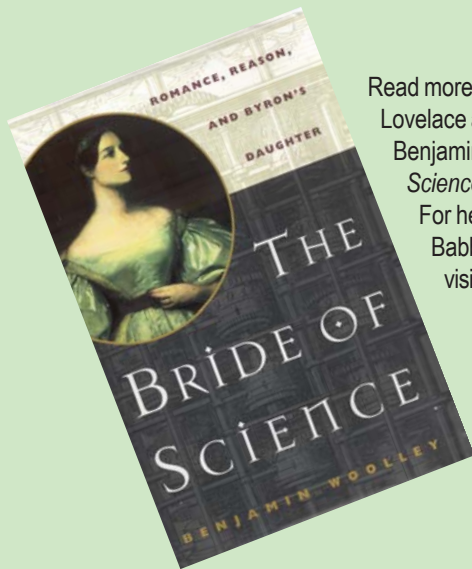
**If you wouldn't insult a customer by using boastful, pompous hype face-to-face or online, why do it offline?**

8. **Lists in the form of long sentences don't get read.** Online, if you have more than two or three items to list you're advised to create bullets, rather than run them together in a long sentence. If that makes them quicker to absorb online, think what a beneficial effect it can have on lists in offline text...
9. **Headlines and crossheads must be relevant, not cutesy-clever.** In the online environment these lines often have to stand alone (e.g., as e-mail subject lines), so must be directly relevant. Although abstract headlines are acceptable in some press ads, in longer offline text the headlines are what people latch on to while scanning. This means they also have to be directly relevant, so they're instantly understood.
10. **Cut the c\*\*\*, get to the point.** Not only do online comms demand uncluttered information, but also relevant information. People haven't got time to wait 10 minutes while your incredibly creative animation downloads, and equally they haven't time to figure out the meaning of a literary quote over an artsy picture when they're in a hurry to find out about your diesel generators. In our high-speed business culture, direct is beautiful. ←

*Suzan St Maur writes extensively on business and marketing communications for publications and Web sites in all the major English language markets. Her latest book, **Powerwriting: the hidden skills you need to transform your business writing is available from Amazon.com. Visit Ms. St Maur at ([www.suzanstmaur.com](http://www.suzanstmaur.com))***

*We reprint Ms. St. Maur's article here with the kind permission of **MarketingProfs.com**, an excellent resource for online-marketing expertise.*

more work after this publication due to a series of illnesses. She succumbed to cancer at the age of 36. In her life, she had foreseen many significant developments in computing, particularly information storage, graphic design, and music analysis and composition, that would not appear until 50 years or more after her death. Her vision and her accomplishments give her a place among a select few who have contributed as much to IT. ☞



Read more about the life of Lady Lovelace and her contributions in Benjamin Woolley's *The Bride of Science* (McGraw-Hill, 2000). For her theories about Charles Babbage's Analytical Engine, visit [www.fourmilab.to](http://www.fourmilab.to)

language did not have an official name. It was called DoD-1, but the HOLWG did not accept this name because the military overtones might be a turn-off for potential non-military users. So the language needed a real name. In May 1979, Commander John Cooper (Navy) suggested using the name Ada, after the British mathematician who was the world's first computer programmer. The HOLWG approved unanimously, and after receiving permission from Ada's descendant, the Earl of Lytton, the new language was approved December 10, 1980, and given the official designation of MIL-STD-1815, in honor of Ada's birthday on December 10, 1815.

In 1987, Ada 83 was released to the public, and in 1995 S. Tucker Taft completed revisions to it and released Ada 95. The language is still used where safety and reliability are crucial and a software failure could be costly, such as space shuttles, Boeing jets, air traffic control equipment, and high-speed trains. It is also often used for teaching beginning programmers, since it is easier to learn than some of its counterparts, such as C++ and Fortran. ☞

## stretch away stress

Comfortable at work? Remember, you are how you sit. At your computer, your head and neck should be upright, and your trunk, shoulders, and upper arms perpendicular to the floor. Your forearms, wrists, and hands should be straight and parallel to the floor. It's also important to vary your tasks and take micro-breaks throughout your day. Getting up for a Pop Tart® or a cup of coffee can actually be good for your health, as are the stress releasers described below.

### Finger Curl

*Benefit: Helps to relax your hands by moving the small muscles of your fingers.*

Start with your hands in front of you, palms facing each other. Straighten your fingers and thumbs, then bend the top two joints of your fingers down toward the top of your palms. Keep the knuckle joint straight. If your fingers are very stiff, bend each finger individually, helping with the other hand. Repeat twice.

### Sit & Reach

*Benefit: Helps relieve tension in your middle back that results from sitting for long periods.*

Move your chair away from your desk so you have room to stretch. Interlock your fingers, turn your palms outward and extend your arms away from your body as far as you can. Keeping your back straight and arms parallel to the desktop, turn your shoulders to the right and return to the center. Turn your shoulders to the left, and then return to the center.

### Face Stretch

*Benefit: Reduces facial tension caused by eyestrain. (Also fewer interruptions from office mates who may now avoid you.)*

Raise your eyebrows and open your eyes as wide as possible. At the same time, open your mouth and stretch the muscles around your nose and chin. Stick your tongue out. Hold for 5–10 seconds.

### One Love

*Benefit: Relieves tension. (Combine with face stretch for even fewer interruptions.)*

Wrap both arms around your body as if to give yourself a hug. Hold for a count of 5, then relax. Repeat 3 times.



## >> DECOUPLING

a Source can come from an existing XML document object, a file, or a generic java.io.InputStream, and the Result can be connected to a new XML document, a file, or a generic java.io.OutputStream. The actual code would look something like:

```
TransformerFactory factory =  
TransformerFactory.newInstance();  
Transformer transformer =  
factory.newTransformer(new DOMSource(myXSL));  
DOMResult result = new DOMResult(myDocument);  
transformer.transform(new StreamSource(new  
File("some_file")), result);
```

This would use the stylesheet stored in myXSL to transform the “some\_file” file and put the result into myDocument in memory.

Microsoft also supplies an XML/XSL processor for its programming languages. To use it in VB, you’d add a project reference to the appropriate version of “Microsoft XML” (I’m using version 3.0, but there is a 4.0 available now). To do the same transformation as in the example above, we’d need code that looks something like:

```
Dim source as new DOMDocument  
Dim myDocument as new DOMDocument  
source.load "some_file"  
myDocument.loadXML source.transformNode(myXSL)
```

The complete documentation for Microsoft’s XML processor is available at <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/xmlsdk30/htm/xmmscxmlreference.asp>.

**THIS CONCLUDES OUR LOOK AT THE DIFFERENT** methods for separating your business logic from your presentation. We’ve seen how the server-side scripting languages (ASP & JSP) practically encourage you to mix your presentation and logic, and we’ve seen how embedded tag technologies start to help you separate it out. But because both of these technologies tend to center you on the presentation, it is fairly easy to become badly coupled. Using XML/XSL to generate your Web pages gives you a very clean implementation of the MVC (model-view-controller) pattern: The XML data is the model, the XSL stylesheet is the view, and the controller is provided by something entirely different. I’m finding that I like using a simple Java servlet—which converts its requests to XML documents and passes them off to simple Java classes to process—as the controller. This allows me to completely separate my presentation (in the form of XSL style sheets) from my business logic. Thus I can put an entirely different front end, such as a Java GUI, without having to change my backend at all. ↵



3535 Canal Street  
New Orleans, LA 70119-6157

PRSRSTD  
US POSTAGE  
PAID  
NEW ORLEANS LA  
PERMIT NO. 2554



Come see us at **XP Agile Universe August 10–13.**  
For complete information visit [www.agileuniverse.com](http://www.agileuniverse.com).